

# Chương 2: Ngôn ngữ lập trình web tĩnh – HTML5

Giảng Viên: ThS. Vũ Minh Sang

# Nội dung



- Ngôn ngữ lập trình web tĩnh HTML 5
- Thành phần mới của HTML5
- Các đối tượng trên Form
- Thuộc tính của Form



# Giới thiệu HTML5

# Giới thiệu HTML5

- Là phiên bản thứ năm của HTML cũng là phiên bản mới nhất hiện nay
- Bản phác thảo bản đầu tiên được công bố vào năm 2008.
- Năm 2014, bản HTML5 của W3C được phát hành.
- Năm 2016, W3C phát hành phiên bản HTML 5.1 Candidate Recommendation

# Đặc điểm HTML5

- Bao gồm các mô hình xử lý chi tiết để tăng tính tương thích, mở rộng, cải thiện và chuẩn hóa các tài liệu trên web.
- Giúp tạo ra các giao diện phức tạp
- Giúp ta tạo ra các ứng dụng web chạy trên thiết bị di động.
- Cung cấp các thẻ, các đối tượng mới giúp hiển thị các đối tượng đa phương tiện đơn giản hiệu quả.
- Cung cấp các thẻ để hiển thị các đối tượng đồ họa đơn giản mà hiệu quả
- Cung cấp các thẻ để tạo ra giao diện một cách đơn giản, dễ nâng cấp và sửa chữa.

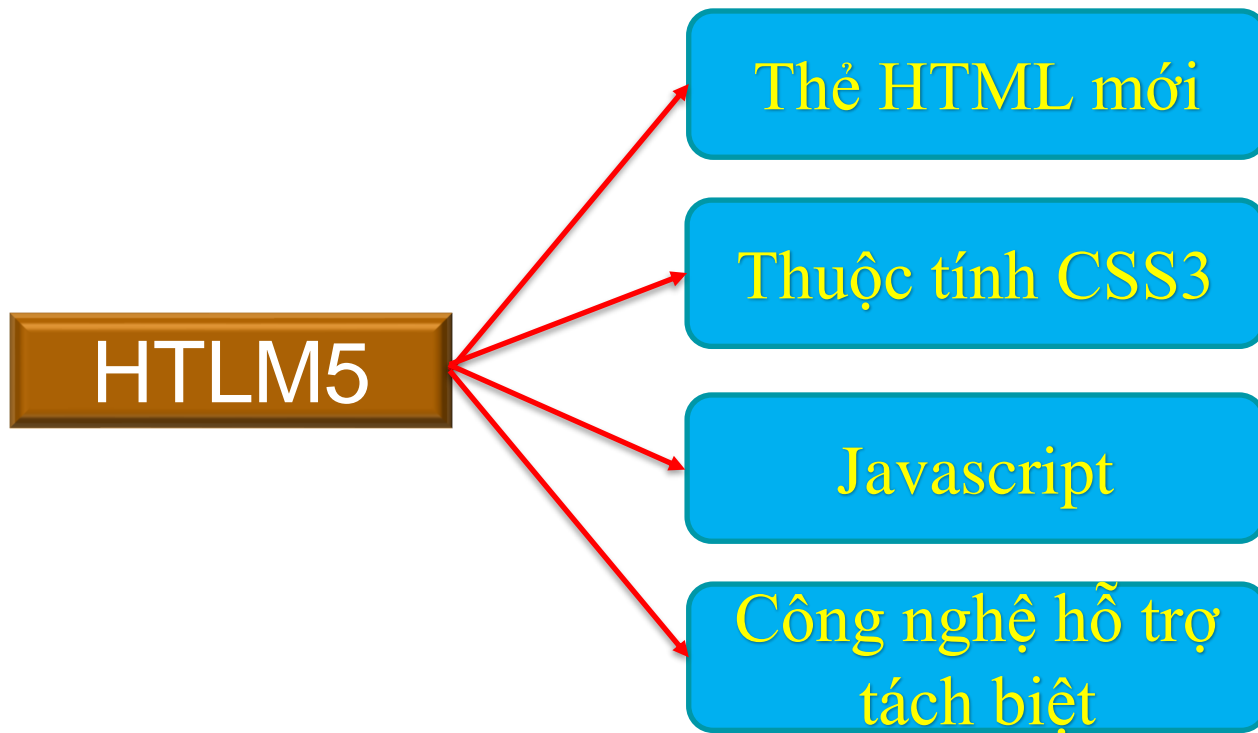
# Đặc điểm HTML5

- Bao gồm các mô hình xử lý chi tiết để tăng tính tương thích, mở rộng, cải thiện và chuẩn hóa các tài liệu trên web.
- Giúp tạo ra các giao diện phức tạp
- Giúp ta tạo ra các ứng dụng web chạy trên thiết bị di động.
- Cung cấp các thẻ, các đối tượng mới giúp hiển thị các đối tượng đa phương tiện đơn giản hiệu quả.
- Cung cấp các thẻ để hiển thị các đối tượng đồ họa đơn giản mà hiệu quả
- Cung cấp các thẻ để tạo ra giao diện một cách đơn giản, dễ nâng cấp và sửa chữa.

# Điểm mạnh của HTML5

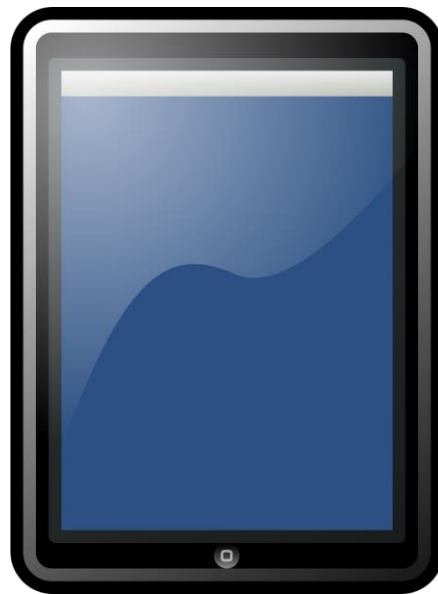
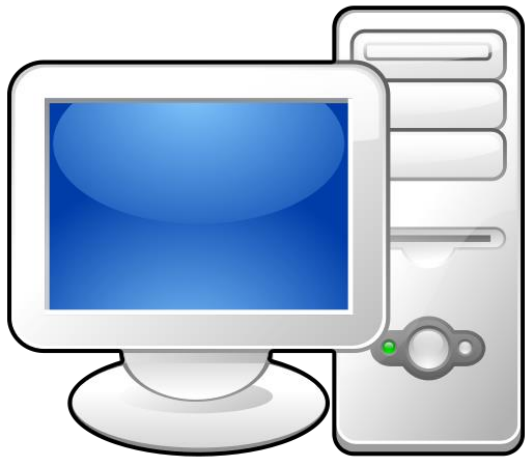
- Khả năng tương thích:
  - Giữ nguyên cấu trúc của các phiên bản trước.
  - Thêm nhiều tính năng mới.
  - Phù hợp trên nhiều nền tảng, khả năng sử dụng trên nhiều hệ thống.
- Tính tiện dụng:
  - Cú pháp HTML5 khá thoải mái, khá đơn giản, thiết kế hỗ trợ sẵn bảo mật
  - Tách biệt giữa phần nội dung và trình bày ngày càng thể hiện rõ.
- Khả năng truy xuất rộng rãi:
  - HTML5 mang lại sự hỗ trợ tốt hơn cho người dùng.
  - Tăng khả năng phục vụ đa phương tiện
  - Hỗ trợ API và DOM.

# Thành phần của HTML5





# Phạm vi sử dụng HTML5



# Tổng quan câu lệnh HTML5

## Phiên bản cũ

`<div id="header">` Đây là header `</div>`

```
#header { width:400px; high:100px;  
background-color:red; }
```

## HTML5

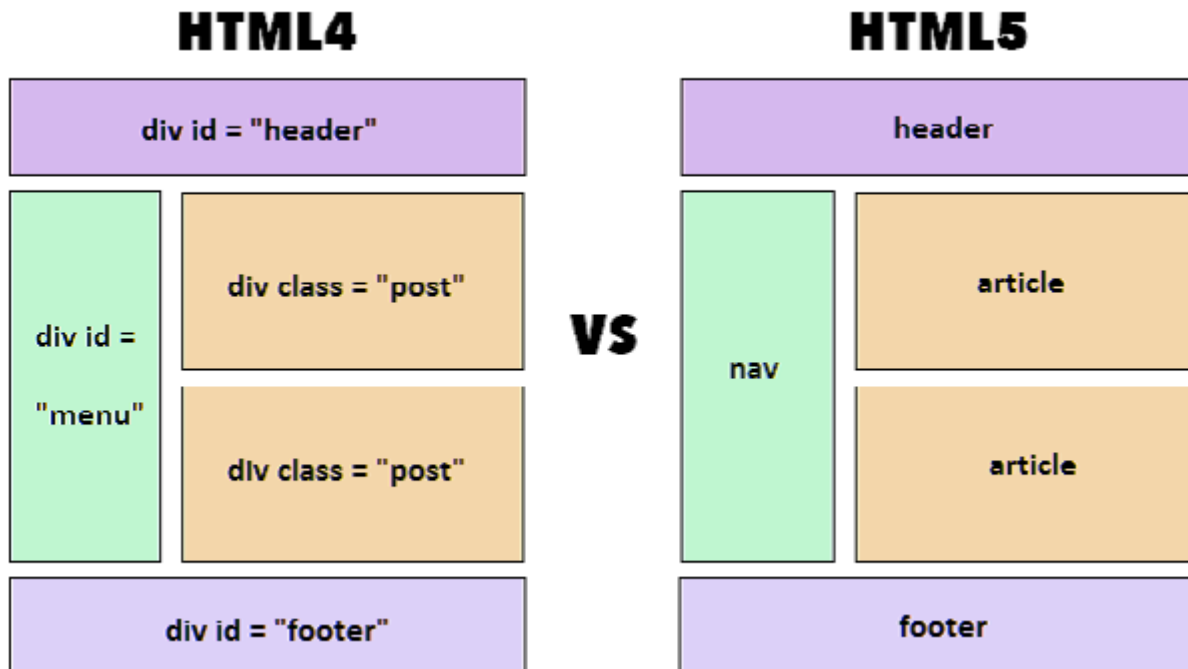
`<header>`Đây là header`</header>`

```
header { width:400px; high:100px;  
background-color:red; }
```

# Tổng quan câu lệnh HTML5

- Tên các thành phần được dựa theo tên các thành phần thông dụng được sử dụng trong phần bố cục trang web hiện nay (div id="header", div id="footer", div id="nav"...)
- Tác dụng của các thành phần này:
  - Giảm bớt sự phụ thuộc vào thẻ <div>
  - Cấu trúc trang web thống nhất, dễ đọc hơn
- HTML **không** thay thế bất kỳ cú pháp HTML nào, chỉ **bổ sung** các thẻ mới.

# Tổng quan câu lệnh HTML5



# Tổng quan câu lệnh HTML5

- Không phân biệt chữ hoa thường
  - `<H1>Tiêu đề</h1>`
- Không bắt buộc phải có thẻ đóng:
  - `<p>Đoạn văn bản`
- Không bắt buộc có dấu nháy kép cho thuộc tính
  - `<img src=impala3.jpg ALIGN =top />`

# Thành phần mới của HTML5

# Cấu trúc file HTML5

**DOCTYPE**



**CONTENT**

**Metadata**

**Flow**

**Sectioning**

**Heading**

**Phrasing**

**Embedded**

**Interactive**

# DOCTYPE

- DOCTYPE được sử dụng để kiểm tra tính hợp lệ của các trang web

## Phiên bản cũ

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD  
HTML 4.0 Transitional//EN"  
"http://www.w3.org/TR/htnl4/loose.dtd"  
>
```

## HTML5

```
<!DOCTYPE HTML>
```



# Content



- Content được chia thành 7 mô hình như sau

## CONTENT

**Metadata**

**Flow**

**Sectioning**

**Heading**

**Phrasing**

**Embedded**

**Interactive**



# Content - Metadata

- Thiết lập cách trình bày hoặc hành vi của các nội dung còn lại trong trang
- Có thể được dùng để thiết lập quan hệ giữa các trang HTML
- Thường chứa các từ khóa mô tả trang web, và được các bộ máy tìm kiếm sử dụng để tìm hoặc phân loại trang.
- Đặt trong phần `<head></head>`
- Gồm các thẻ: `base`, `command`, `link`, `meta`, `noscript`, `script`, `style`, `title`

```
<head>
<title>Page Title</title>
<meta charset=utf-8 />
<link rel="stylesheet" type="text/css" href="style.css" />
<script type="text/javascript" src="scripts.js"></script>
</head>
```

# Content - Flow

- Chứa tất cả các thẻ tạo luồng nội dung trong trang web: `a`, `b`, `blockquote`, `div`, `em`, `h1`, `h2`...
- Một số thẻ mới trong HTML5: `article`, `aside`, `audio`, `canvas`, `hgroup`

```
<h1>My home page</h1>
```

```
<p>I like playing with string, I guess. Sister says squirrels are  
fun too so sometimes I follow her to play with them.</p>
```

# Content - Sectioning

- Trình bày phạm vi nội dung trong các headings, footers, điều hướng...
- Gồm các thẻ: `article`, `aside`, `nav`, `section`

```
<article>
  <h1>Tag moi</h1>
  <p>Hoc lap trinh Web</p>
</article>
```

# Content - Heading

- Trình bày các tiêu đề nội dung (header) của website gồm các thẻ từ `h1` đến `h6`
- Thêm một thẻ mới: `hgroup`

```
<hgroup>  
  <h1>Main Heading</h1>  
  <h2>Sub-heading</h2>  
</hgroup>
```

# Content – Phrasing

- Chứa thẻ đánh dấu văn bản, các đoạn văn, ...
- Gồm các thẻ: `a`, `abbr`, `area`, `audio`, `b`, `bdo`, `br`, `button`, `var`, `sub`, `sup`, `strong`, `wbr`,...

```
<p><em>A locally owned</em> and  
<strong>independent</strong> widget store since 1965<br/>  
</p>
```

# Content – Embedded

- Chứa nội dung và import những sources khác vào văn bản.
- Gồm các thẻ: `audio`, `canvas`, `embed`, `iframe`, `img`, `math`, `object`, `svg`, `video`

```
<audio controls>  
  <source src="horse.ogg" type="audio/ogg">  
  <source src="horse.mp3" type="audio/mpeg">  
</audio>
```

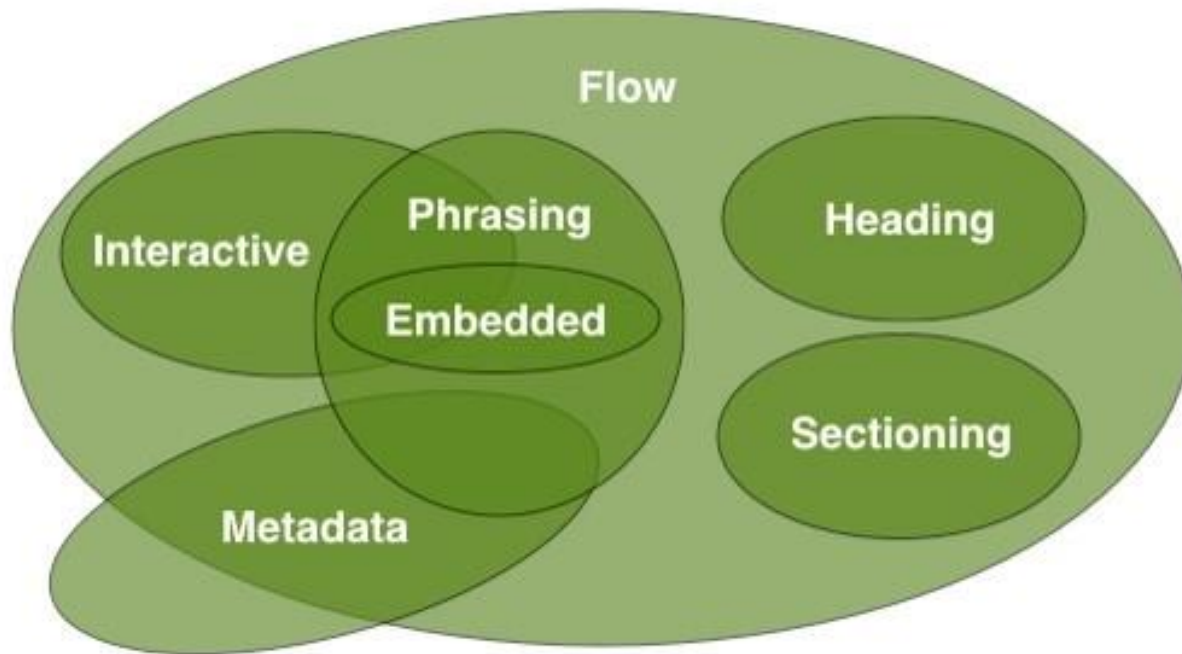
# Content – Interactive

- Chứa những thẻ đặc biệt cho sự tương tác của người dùng
- Gồm các thẻ: `a`, `audio`, `button`, `details`, `embed`, `img`, `input`, `keygen`, `lable`, `menu`, `object`, `select`, `textarea`, `video`

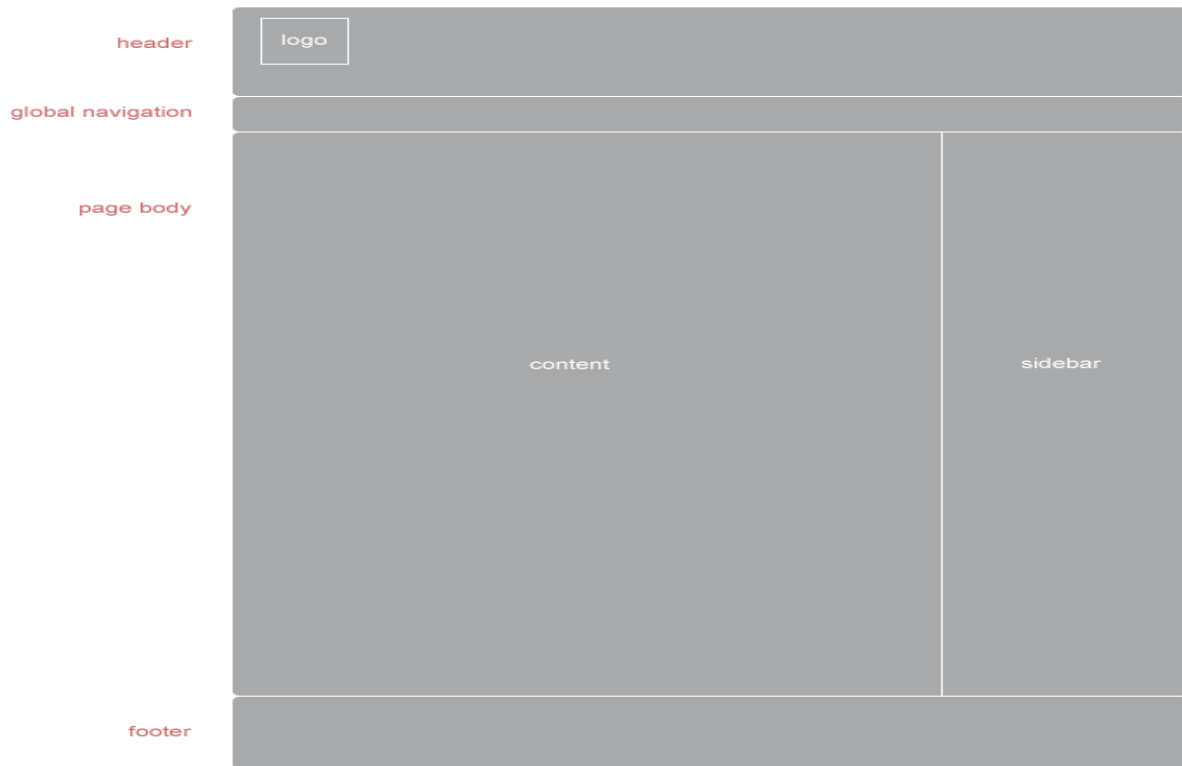
```
<input type="button" value="Click" onClick='alert("Hello")' />
```



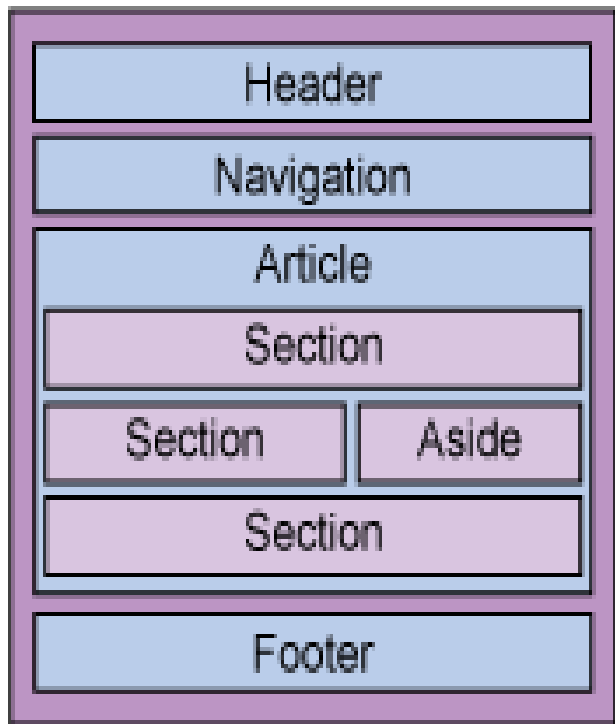
# Content



# Cấu trúc trang web



# Các thành phần trong trang HTML5



# Phần Header

- Chứa tiêu đề và phụ đề trang, các tag từ h1 đến h6
- Có thể chứa thông tin mở về một <section> và <article> ngoài chính trang web đó.
- Có thể dùng nhiều lần trong một trang web

```
<header>
```

```
  <h1>Heading Text</h1>
```

```
  <p> Text or image can be included here</p>
```

```
  <p> Logos are frequently placed here too</p>
```

```
</header>
```

# Phần Header – Thẻ Hgroup

- Chứa tiêu đề và phụ đề trang, các tag từ h1 đến h6
- Thẻ `<header>` cũng có thể chứa một thẻ `<hgroup>` dùng để tạo nhóm các tiêu đề với nhau

```
<header>
  <hgroup>
    <h1>Main Heading</h1>
    <h2>Sub-heading </h2>
  </hgroup>
  <p> Text or images can be included here</p>
</header>
```

# Phần Navigation

- Chứa các thành phần dùng để chuyển hướng trang trong trang web
- Dùng thẻ `<nav>`

```
<nav>
```

```
  <ul>
```

```
    <li><a href="#">Home</a></li>
```

```
    <li><a href="#">About Us</a></li>
```

```
    <li><a href="#">Our Products</a></li>
```

```
    <li><a href="#">Contact Us</a></li>
```

```
  </ul>
```

```
</nav>
```

# Phần Section

- Dùng để hiển thị một vùng chung của trang
- Sử dụng Section khi muốn phân chia nội dung quan trọng như văn bản hoặc hình ảnh thành các vùng

```
<h1>Product User Guide</h1>
```

```
<section>
```

```
  <h2>Setting it Up</h2>
```

```
</section>
```

```
<section>
```

```
  <h2>Basic Features</h2>
```

```
  <section>
```

```
    <h2>Video Playback</h2>
```

```
  </section>
```

```
</section>
```

# Phần Article

- Thể hiện một phần độc lập của site, có thể bao gồm nhiều Section và ngược lại.
- Phần article có thể có một `<header>`, `<footer>`

```
<article>
  <h1>Title article</h1>
  <section> Content </section>
  <section> Content </section>
</article>
<section>
  <h1>Title section</h1>
  <article> Content </article>
  <article> Content </article>
</section>
```



# Phần Aside

- Dùng thể hiện những phần trong khung nội dung nhưng được cố định không thay đổi như các menu phụ ở cột right, left

```
<aside>
```

```
  <h4>Disney in France</h4>
```

```
  <p>Besides Euro Disney, there is a Disneyland in California</p>
```

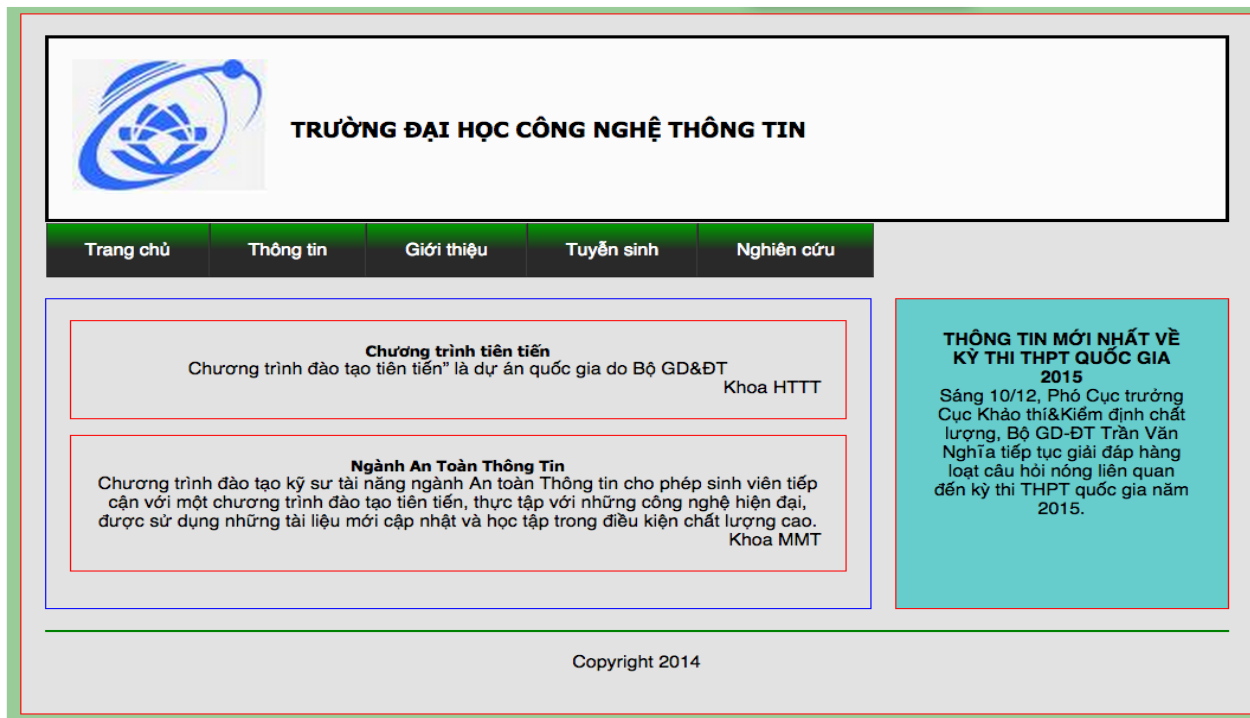
```
</aside>
```

## Phần Footer

- Được dùng để thể hiện nội dung ở phần cuối trang.
- Thường thì nội dung của phần này sẽ chứa các thông tin tác giả, bản quyền, liên hệ...

```
<footer>  
    <p>&copy; Copyright 2020. All rights reserved</p>  
</footer>
```

# Ví dụ



# Form trong HTML5

# Giới thiệu Form

- Chức năng giống với các phiên bản HTML cũ
- Các thành phần mới của form trong HTML5 bổ sung một số chức năng mà không cần thông qua các phương tiện khác như javascript
- Cú pháp:

```
<form id="..." action="..." method="...">  
    <!-- các thành phần của Form -->  
</form>
```

- **id** : cho phép định dạng form với CSS
- **action** : chỉ định trang web nhận xử lý dữ liệu từ FORM này khi có sự kiện click của button SUBMIT.
- **method** : Xác định phương thức chuyển dữ liệu (POST,GET)

# Giới thiệu Form

- HTML5 hỗ trợ Web Form 2.0 tạo ra các form mạnh mẽ hơn
- Các điều khiển chọn ngày tháng, màu sắc, chọn số.
- Các kiểu input mới: email, tel, search, url...
- Ngoài phương thức get, post còn có thêm các phương thức put và delete

# Các phần tử của Form HTML5 mới

# Các loại Input mới

- HTML5 có một số loại **input** mới cho các hình thức. Những tính năng mới cho phép kiểm soát đầu vào và xác nhận tốt hơn.
- Các loại input mới như: color, date, time, email, url,.....

- Dạng color: nhập màu sắc

```
<input type="color" name="" />
```

- Dạng date: nhập ngày tháng

```
<input type="date" name="" />
```

- Dạng datetime

```
<input type="datetime" name="" />
```

- Dạng datetime-local

```
<input type="datetime-local" name="" />
```



# Các loại Input mới

- Dạng month: chọn ngày tháng  
`<input type="month" name="" />`
- Dạng week: chọn tuần  
`<input type="week" name="" />`
- Dạng time: chọn thời gian  
`<input type="time" name="" />`
- Dạng email: nhập email  
`<input type="email" name="" />`
- Dạng url: nhập địa chỉ URL  
`<input type="url" name="" />`

# Các loại Input mới

- Dạng search: hộp tìm kiếm

```
<input type="search" name="" />
```

- Dạng tel: nhập số điện thoại

```
<input type="tel" name="" />
```

- Dạng number: nhập số

```
<input type="number" name="" />
```

- Dạng range: nhập dải số.

```
<input type="range" name="" />
```

# Các thuộc tính mới của Form HTML5

# Autocomplete

- Thuộc tính **autocomplete** cho phép trình duyệt sẽ tự động lưu lại các giá trị mà người dùng nhập trên web trước đó.
- Được thiết lập với 2 giá trị **on** hoặc **off**

```
<form action="demo_form.php" autocomplete="on">  
  First name:<input type="text" name="fname"><br>  
  Last name: <input type="text" name="lname"><br>  
  E-mail: <input type="email" name="email" autocomplete = "on"><br>  
<input type="submit">  
</form>
```

# Autofocus

- Thuộc tính **autofocus** được khai báo, khi trang web được load, thì con trỏ sẽ nằm ở vị trí input được khai báo.
- Các trình duyệt hỗ trợ :Opera, Chrome, Firefox, Safari

First name : `<input type="text" name="fname" autofocus>`

# Novalidate

- Thuộc tính **novalidate** làm cho dữ liệu trong input không cần được kiểm tra đúng kiểu trước khi gửi đi.
- Thuộc tính novalidate là thuộc tính Boolean.
- Các trình duyệt hỗ trợ: Opera, Chrome, Firefox

```
<form action="demo_form.php" novalidate >  
    First name:<input type="text" name="fname"><br>  
    E-mail: <input type="email" name="email"><br>  
    <input type="submit">  
</form>
```

# Thuộc tính của thẻ input - form

- Thuộc tính **form** được khai báo để cho biết các input thuộc form nào.
- Các trình duyệt hỗ trợ: Opera, Chrome, Firefox, Safari

```
<form action="demo_form.php" id="form1">  
    First name: <input type="text" name="fname"><br>  
    <input type="submit" value="Submit">  
</form> Last name:<input type="text" name="lname" form="form1">
```

# Thuộc tính của thẻ input – formaction

- Thuộc tính **formaction** giúp chúng ta có thể tùy chọn file nào sẽ xử lý dữ liệu nhập vào. Thuộc tính **formaction** sẽ ghi đè lên thuộc tính **action** của thẻ form
- Các trình duyệt hỗ trợ: Opeara, Chrome, Firefox, Safari

```
<form action="demo_form.asp" method="get">  
    First name: <input type="text" name="fname"><br>  
    Last name: <input type="text" name="lname"><br>  
    <input type="submit" value="Submit">  
    <input type="submit" formaction="demo_admin.php"  
        value="Submit using POST">  
</form>
```



# Thuộc tính của thẻ input – formmethod

- Thuộc tính **formmethod** quy định phương thức gửi form đến phần xử lý trên sever. Thuộc tính **formmethod** ghi đè lên thuộc tính **method** của phần tử form.
- Các trình duyệt hỗ trợ: Opeara, Chrome, Firefox, Safari

```
<form action="demo_form.asp" method="get">  
    First name: <input type="text" name="fname"><br>  
    Last name: <input type="text" name="lname"><br>  
    <input type="submit" value="Submit">  
    <input type="submit" formmethod="post" value="Submit using POST">  
</form>
```

# Thuộc tính của thẻ input – formenctype

- Thuộc tính **formenctype** giúp chúng mã hóa dữ liệu “**multipart/form-data**” khi gửi đến server. Thuộc tính **formenctype** sẽ ghi đè lên thuộc tính **enctype** của phần tử <form>.
- Các trình duyệt hỗ trợ: Opeara, Chrome, Firefox, Safari

```
<form action="demo_post_enctype.asp" method="post">  
  First name: <input type="text" name="fname"><br>  
  <input type="submit" value="Submit">  
  <input type="submit" formenctype="multipart/form-data"  
    value="Submit as Multipart/form-data">  
</form>
```

# Thuộc tính của thẻ input – formnovalidate

- Thuộc tính **formnovalidate** làm cho form không bắt lỗi dữ liệu người dùng nhập vào. Thuộc tính **formnovalidate** sẽ ghi đè lên thuộc tính **novalidate** của <form>.
- Các trình duyệt hỗ trợ:Opera, Chrome, Firefox, Safari

```
<form action="demo_form.asp">  
  E-mail: <input type="email" name="userid"><br>  
  <input type="submit" value="Submit"><br>  
  <input type="submit" formnovalidate="formnovalidate"  
  value="Submit without validation">  
</form>
```

# Thuộc tính của thẻ input – formtarget

- Thuộc tính **formtarget** được khai báo tùy theo giá trị của **target** sau khi submit dữ liệu thì sẽ mở trang mới hoặc cửa sổ mới,..  
Thuộc tính **formtarget** ghi đè thuộc tính **target** của <form>
- Các trình duyệt hỗ trợ: Opeara, Chrome, Firefox, Safari

```
<form action="demo_form.asp">
  First name: <input type="text" name="fname"><br>
  Last name: <input type="text" name="lname"><br>
  <input type="submit" value="Submit as normal">
  <input type="submit" formtarget="_blank"
  value="Submit to a new window">
</form>
```

# Thuộc tính min - max

- Thuộc tính **min** và **max** xác định giá trị nhỏ nhất và lớn nhất của thẻ `<input>`.
- Các trình duyệt hỗ trợ: Opera, Chrome, Firefox, Safari

Nhập thời gian sau ngày 2000-01-01:

```
<input type="date" name="bday" min="2000-01-02"><br>
```

Số lượng (từ 1 đến 5):

```
<input type="number" name="quantity" min="1" max="5"><br>
```

# Thuộc tính step

- Thuộc tính **step** xác định bước nhảy số của thẻ `<input>` kiểu số.

- Input dạng number

```
<input type="number" name="" min="3" max="10" step="5" />
```

- Input dạng range

```
<input type="range" name="" min="1" max="100" step="10" />
```

# Thuộc tính của input – pattern, placeholder, required

- Thuộc tính **pattern** tạo ra sự ràng buộc đối với dữ liệu nhập vào. Các trình duyệt hỗ trợ: Opera, Chrome, Firefox, Safari.
- Thuộc tính **placeholder** tạo ra giá trị mẫu trên thẻ **<input>**, giá trị mẫu sẽ bị xóa đi nếu ta chọn đến trường của nó. Các trình duyệt hỗ trợ: Opera, Chrome, Firefox, Safari.
- Thuộc tính **required** có 2 giá trị **có** hoặc **không**. Thuộc tính này quy định một trường nhập liệu bắt buộc không được rỗng. Các trình duyệt hỗ trợ: Opera, Chrome, Firefox.

```
<input type="text" name="username" id="username" placeholder="4 <> 10"
pattern="[A-Za-z]{4,10}" required >
```

# Datalist

- Định nghĩa một danh sách tùy chọn, sử dụng `<input>` xác định giá trị chọn từ danh sách.

```
<input list="browsers">
```

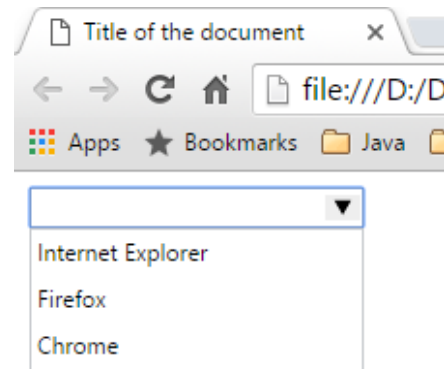
```
<datalist id="browsers">
```

```
<option value="Internet Explorer">Internet Explorer</option>
```

```
<option value="Firefox">Firefox</option>
```

```
<option value="Chrome">Chrome</option>
```

```
</datalist>
```

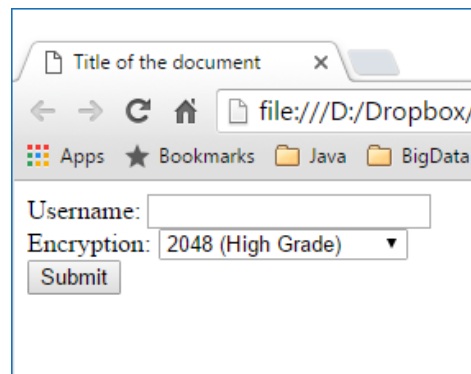




# Tag <keygen>

- Tạo sự bảo mật cho xác thực người dùng. Nó sẽ định ra một cặp khóa trong form. Khi ta nhấn chọn gửi dữ liệu, hai mã khóa được tạo ra, một khóa **private** và một khóa **public**.

```
<form action="demo_keygen.asp" method="get">
  Username: <input type="text" name="usr_name">
  Encryption: <keygen name="security">
    <input type="submit">
</form>
```

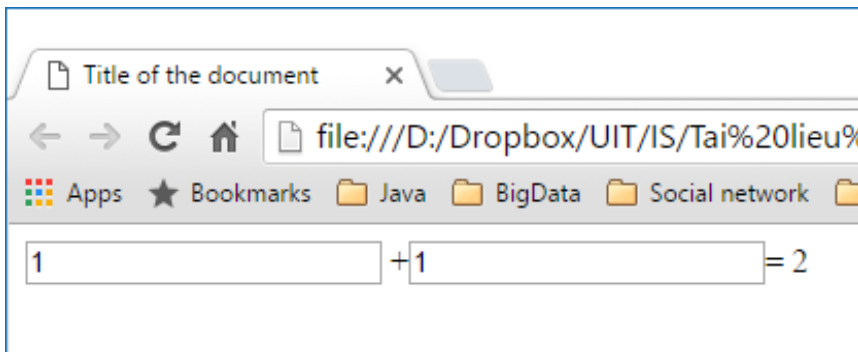


The screenshot shows a web browser window with a single tab titled 'Title of the document'. The address bar shows 'file:///D:/Dropbox/'. The browser has a toolbar with 'Apps', 'Bookmarks', 'Java', and 'BigData' buttons. The form contains a 'Username:' label followed by a text input field, an 'Encryption:' label followed by a dropdown menu currently showing '2048 (High Grade)', and a 'Submit' button.

# Tag <output>

- Tượng trưng cho một kết quả của một phép tính.

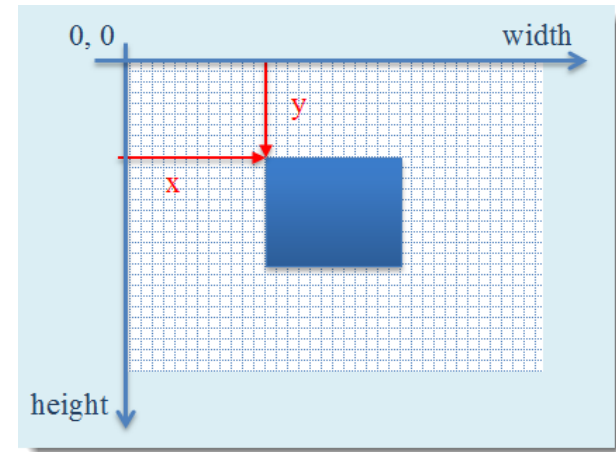
```
<form oninput="x.value=parseInt(a.value)+parseInt(b.value)">
  <input type="text" name="a">+<input type="text" name="b">=
  <output name="x" for="a b"> </output>
</form>
```



# Graphics (Canvas, SVG)

# Canvas

- Canvas HTML5 là một **API vẽ 2 chiều** và tạo hình ảnh động, sử dụng JavaScript để vẽ đồ họa trực tiếp trên trang web.
- Canvas HTML5 là tag lý tưởng để xây dựng các hình ảnh thú vị, các bản vẽ, các album ảnh, các biểu đồ, các đồ thị, các hình ảnh động, và các ứng dụng bản vẽ nhúng.
- Một canvas có thể được coi như một lưới với số pixel được xác định bởi độ rộng width và độ cao height. Tọa độ trong canvas có gốc nằm ở góc trên trái



# Canvas

- Các bước làm việc với canvas

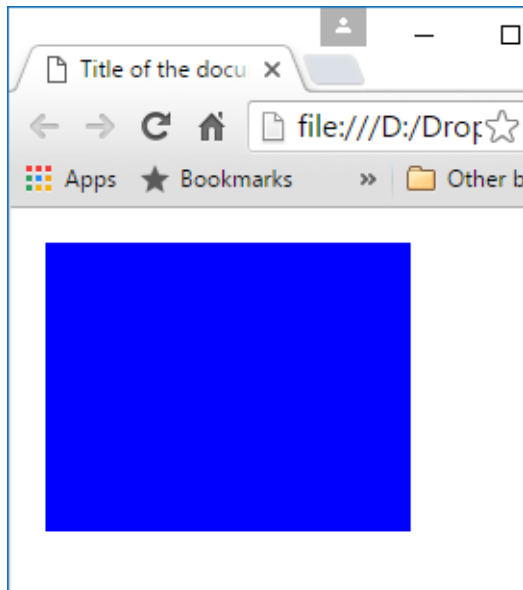
- Định nghĩa thành phần canvas trong html:

```
<canvas id="myCanvas" width="200" height="200"></canvas>
```

- **id:** mã định danh
- **width:** chiều rộng
- **height:** chiều dài
- **class:** lớp
- **name:** tên

- Tham chiếu bối cảnh vẽ cho các phần tử đó như một biến trong javascript

```
<script type="text/javascript">
    var c=document.getElementById("myCanvas");
    var cont=c.getContext("2d");
    cont.fillStyle="blue";
    cont.fillRect(10,10,200,150);
</script>
```



# Canvas

- Javascript xác định đối tượng canvas thích hợp, sử dụng phương thức `document.getElementById()`  

```
var canvas = document.getElementById("myCanvas");
```
- Mỗi phần tử canvas phải có một định nghĩa ngữ cảnh. Hiện nay, đặc tả chính thức chỉ công nhận môi trường 2D  

```
var context = canvas.getContext("2d");
```
- Các công cụ vẽ, các hiệu ứng, và các phép biến đổi.
  - **Công cụ vẽ:** các đường kẻ, các hình chữ nhật, các cung tròn, các đường cong Bezier và đường cong bậc hai, các hình tròn và bán nguyệt.
  - **Các hiệu ứng:** Tô đầy và các nét, các gradient tuyến tính và xuyên tâm.
  - **Các phép biến đổi:** Tỷ lệ, quay tròn, tịnh tiến.

# Canvas

- Ưu điểm của thành phần canvas
  - Cho phép tạo graphic, hình động (animation), gradient, các đối tượng đồ họa bằng mã
  - Được hỗ trợ bởi các trình duyệt phổ biến
  - Khả năng mạnh mẽ: làm game, animation, chart, graph, vector...
  - Không phải sử dụng thêm plugin

# Canvas

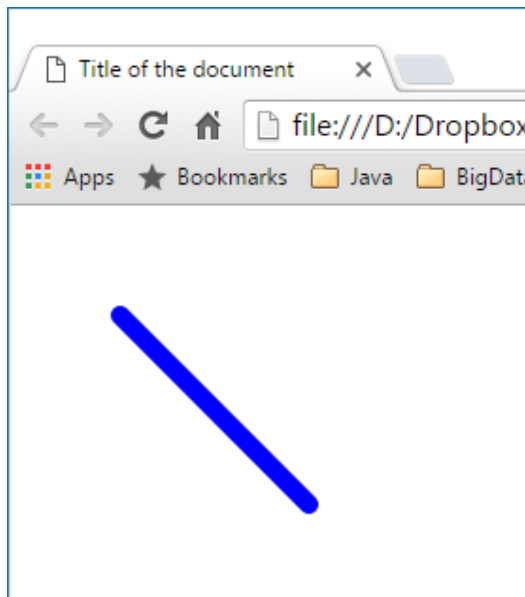
- Các đường vẽ nét sử dụng phương thức:
  - o `context.beginPath();` Bắt đầu một đường kẻ mới
  - o `context.moveTo(x,y);` Nơi bắt đầu đường kẻ mới
  - o `context.lineTo(x,y);` Xác định điểm cuối cho đường vẽ
  - o `context.stroke();` Xác lập vẽ đường kẻ
  - o `context.lineWidth=x;` Xác định độ rộng của đường kẻ
  - o `context.strokeStyle=x;` Màu sắc của đường kẻ
  - o `context.lineCap="x";` Làm tròn các đầu kẻ



# Canvas



```
<script>
  window.onload = function() {
    var canvas = document.getElementById("myCanvas");
    var context = canvas.getContext("2d");
    context.beginPath();
    context.moveTo(150, 150);
    context.lineTo(50,50);
    context.lineWidth = 10;
    context.strokeStyle = "#0000FF";
    context.lineCap = "round";
    context.stroke();
  };
</script>
</head>
<body>
  <canvas id="myCanvas" width="400" height="200"></canvas>
</body>
```



# Canvas

- Vẽ hình chữ nhật sử dụng phương thức:
  - `context.fillRect(x,y,width,height);` Vẽ một hình chữ nhật
  - `context.strokeRect(x,y,width,height);` Vẽ nét ngoài của hình chữ nhật
  - `context.clearRect(x,y,width,height);` Làm trong suốt hình chữ nhật
    - **x, y:** Tọa độ nơi bắt đầu hình chữ nhật sẽ được vẽ
    - **width, height:** chiều dài và chiều rộng tương ứng của hình chữ nhật

# Canvas



```
<script>
    window.onload = function() {
        var canvas = document.getElementById("myCanvas");
        var context = canvas.getContext("2d");
        context.strokeRect(20,20,150,50);
        context.fillRect(20,80,250,50);
        context.clearRect(22,82,230,30);
    };
</script>
</head>
<body>
    <canvas id="myCanvas" width="400" height="200">
    </canvas>
</body>
```



# Canvas

- Vẽ hình tròn, hình bán nguyệt:
  - `context.arc(x, y, r, start, stop, anticlockwise);`
    - **x, y**: Tọa độ tâm hình tròn.
    - **r**: Bán kính hình tròn.
    - **start, stop**: Điểm đầu và điểm cuối của cung tròn, tính bằng radian.
    - **anticlockwise**: (Mặc định là `False`) là một giá trị Boolean. Giá trị **True** vẽ ngược chiều kim đồng hồ. **False** cùng chiều kim đồng hồ.
- Sau khi xác định tham số, dùng một phương thức vẽ nét để vẽ `context.stroke();`

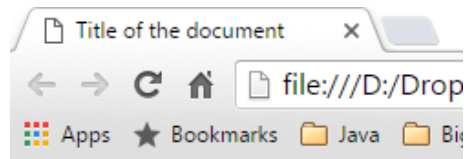
# Canvas



```
<script>
```

```
    window.onload = function() {  
        var canvas = document.getElementById("myCanvas");  
        var context = canvas.getContext("2d");  
        context.arc(100,100,75,0,Math.PI);  
        context.stroke();  
    };
```

```
</script>
```



# Canvas

- Vẽ đường cong bậc 2:
  - `context.moveTo(x, y);`
  - `context.quadraticCurveTo(controlX, controlY, endX, endY)`
    - x, y: **Tọa độ điểm xuất phát.**
    - controlX, controlY : **Tọa độ điểm điều khiển.**
    - endX, endY : **Tọa độ điểm kết thúc.**
- Vẽ đường cong Bezier.
  - `context.moveTo(x, y);`
  - `context.bezierCurveTo(controlX1, controlY1, controlX2, controlY2, endX, endY )`

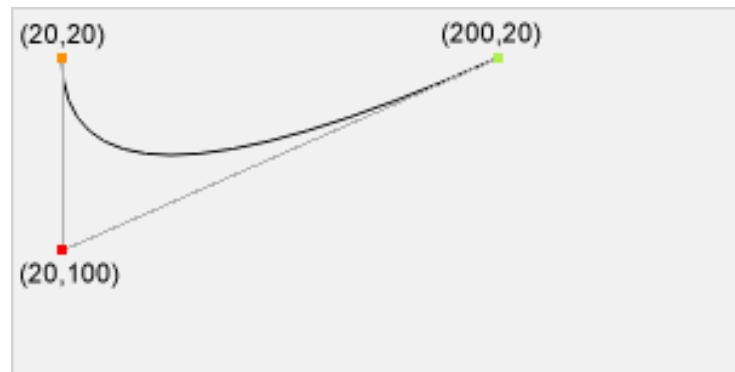
# Canvas



```
<script>
```

```
    window.onload = function() {  
        var canvas = document.getElementById("myCanvas");  
        var context = canvas.getContext("2d");  
        context.moveTo(20, 20);  
        context.quadraticCurveTo(20,100,200,20);  
        context.strokeStyle = "red";  
        context.stroke();  
    };
```

```
</script>
```



# Canvas

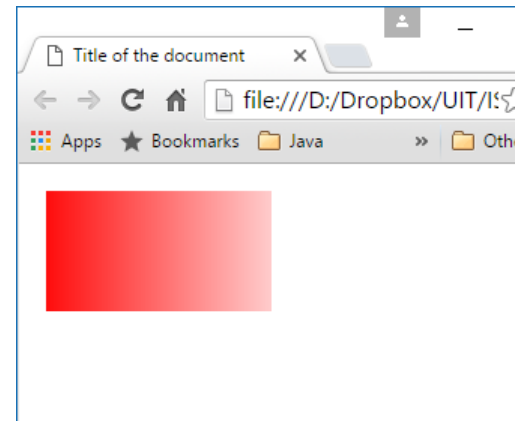
- Canvas Gradient: Một *gradient* là một vùng được tô đầy và di chuyển từ một màu này sang màu khác, pha trộn các màu ở nơi chúng giao nhau.
  - `createLinearGradient(x0,y0,x1,y1);` Quét màu xuất phát từ một phía.
  - `createRadialGradient(x0,y0,r0,x1,y1,r1);` Quét màu từ tâm
    - `x0, y0): (x1, y1):` Tọa độ điểm đầu.Tọa độ điểm cuối.
    - `x0, y0 ,r0 :` Tọa độ, bán kính của đường tròn đầu tiên.
    - `x1, y1, r1 :` Tọa độ, bán kính của đường tròn thứ hai.
- SVG Gradient: Phương thức: `addcolorStop(offset, color)` Xác định màu sẽ được lưu lại ở những giá trị bù cụ thể.
  - `offset:` Giá trị bù (0 or 1) .
  - `color :` Màu ở giá trị bù.
- Phương thức: `fillStyle( )`: hiển thị gradient.



# Canvas



```
<script>
window.onload = function() {
    var canvas=document.getElementById("myCanvas");
    var context=canvas.getContext("2d"); // Create gradient
    var grd=context.createLinearGradient(0,0,200,0);
    grd.addColorStop(0,'red');
    grd.addColorStop(1,'white');
    context.fillStyle=grd;
    context.fillRect(10,10,150,80);
};
</script>
```



# SVG (Scalable Vector Graphics)

- SVG được sử dụng để xác định các đồ họa vector cho Web
- SVG định nghĩa đồ họa ở định dạng XML
- SVG đồ họa hình ảnh không ảnh hưởng đến chất lượng khi chúng được phóng to hoặc thay đổi kích thước ảnh

# SVG (Scalable Vector Graphics)

- Ưu điểm của SVG:
  - Hình ảnh SVG có thể được tạo ra và chỉnh sửa trên trình soạn thảo văn bản.
  - Hình ảnh SVG có thể được tìm kiếm, lập chỉ mục, kịch bản , và nén, hình ảnh SVG được mở rộng
  - Hình ảnh SVG có thể được in với chất lượng cao ở độ phân giải bất kỳ. Có khả năng phóng to mở rộng mà không bị vỡ hình.
  - Hình ảnh SVG Zoomable (và hình ảnh có thể được zoom mà không làm giảm đi chất lượng màu sắc)
  - Các trình duyệt hỗ trợ: Internet Explorer 9+, Firefox, Opera, Chrome, và Safari...

# SVG (Scalable Vector Graphics)

- SVG Shape

- **SGV Shape** có nhiều đối tượng : SGV Circle, Rectangle, Ellipse, Line, Polygon, Polyline, Path, Stroking

- **Ví dụ: SVG Circle**

```
<svg>  
  <circle cx="100" cy="50" r="40" stroke="black" stroke-width="2" fill="red" />  
</svg>
```

- **Ví dụ: SVG Eclipse**

```
<svg>  
  <ellipse cx="300" cy="80" rx="100" ry="50"  
    style="fill:yellow;stroke:purple;stroke-width:5" />  
</svg>
```

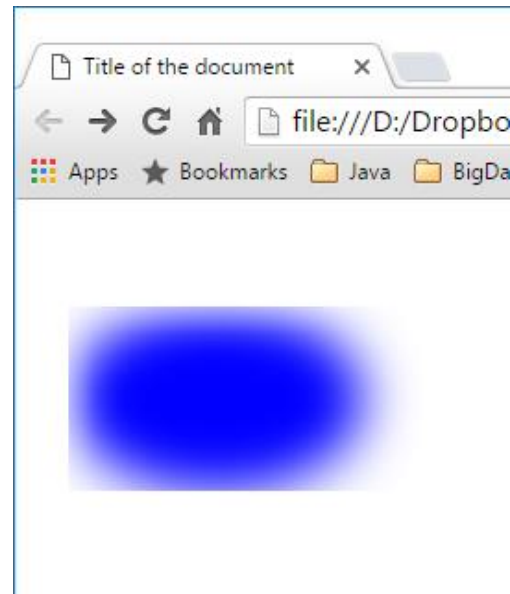
# SVG (Scalable Vector Graphics)

- SVG Filter: Dùng để tạo các hiệu ứng mờ cho ảnh

- Ví dụ:

```
<svg>
  <filter id="f1" x="0" y="0">
    <feGaussianBlur in="SourceGraphic"
      stdDeviation="15" />
  </filter>
  <ellipse cx="300" cy="80" rx="100" ry="50"

    style="fill:blue;stroke:purple;stroke-
      width:0;filter:url(#f1)" />
</svg>
```

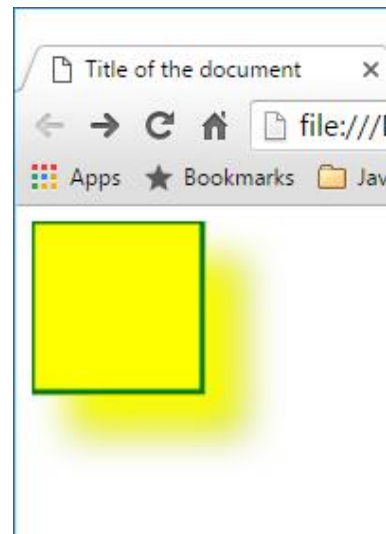


# SVG (Scalable Vector Graphics)

- SVG <feOffset>: Các yếu tố <feOffset> được sử dụng để tạo ra các hiệu ứng đổ bóng

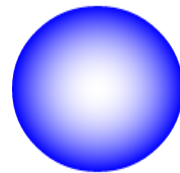
- Ví dụ:

```
<svg>
<defs>
<filter id="f1" x="0" y="0" width="200%" height="200%">
<feOffset result="offOut" in="SourceGraphic" dx="20" dy="20"/>
<feGaussianBlur result="blurOut" in="offOut" stdDeviation="10"/>
<feBlend in="SourceGraphic" in2="blurOut" mode="normal" />
</filter>
</defs>
<rect width="90" height="90" stroke="green" stroke-width="3"
  fill="yellow" filter="url(#f1)" />
</svg>
```



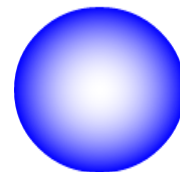
# SVG (Scalable Vector Graphics)

- SVG Gradient: Gradient là một quá trình chuyển đổi từ một màu sang màu khác.
- Có 2 loại SVG Gradients: linear Gradients và radial Gradients
  - SVG `<linearGradients>` : Chuyển màu theo độ dốc ngang, dọc, hay nghiêng
  - SVG `<radialGradients>` : Chuyển màu theo đường dốc xuyên tâm



# SVG (Scalable Vector Graphics)

```
<svg>
  <defs>
    <linearGradient id="grad1" x1="0%" y1="0%" x2="100%" y2="0%">
      <stop offset="0%" style="stop-color:rgb(255,255,0);stop-
opacity:1" />
      <stop offset="100%" style="stop-color:rgb(255,0,0);stop-
opacity:1" />
    </linearGradient>
  </defs>
  <ellipse cx="200" cy="70" rx="85" ry="55" fill="url(#grad1)" />
</svg>
```





# Drag và drop

- Cho phép kéo thả một đối tượng từ vị trí này sang vị trí khác
- Các trình duyệt hỗ trợ: Internet Explorer 9+, Firefox, Opera, Chrome, and Safari

# Multimedia

# Multimedia (Tag Video)

- Nhúng một video vào web mà không cần hỗ trợ Plugin.

```
<video>  
    <source src="E:\Demo.mp4" type="video/mp4">  
    <source src="E:\Demo.org" type="video/org">  
</video>
```

- **"E:\Demo.mp4"** : Đường dẫn đến video.
- Thẻ `<video>` có thể có nhiều thẻ `<source>`, `<source>` có thể chứa nhiều video khác nhau.
- Hỗ trợ 3 định dạng video: mp4, org, webm
- Cần được bổ sung các thuộc tính cho Video, để có thể chạy, điều khiển,...được video.

# Multimedia (Tag Video)

- Thuộc tính kỹ thuật của video

Thuộc tính	Mô tả
<b>width</b>	Thuộc tính này xác định chiều rộng của vùng hiển thị của video, theo pixel CSS.
<b>loop</b>	Thuộc tính boolean này nếu được chỉ định, sẽ cho phép video tự động lặp trở lại bắt đầu sau khi kết thúc
<b>preload</b>	Thuộc tính này xác định rằng video sẽ được nạp lúc tải trang, và sẵn sàng để chạy. Bỏ qua nếu autoplay được chỉ định.
<b>poster</b>	Đây là một URL của một hình ảnh để hiển thị cho đến khi người sử dụng đóng hoặc chuyển.
<b>src</b>	URL của video để nhúng. Đây là tùy chọn, thay vào đó bạn có thể sử dụng các thẻ <source> trong khối video để xác định video để nhúng

# Multimedia (Tag Video)

- Thuộc tính kỹ thuật của video

Thuộc tính	Mô tả
<b>autoplay</b>	Thuộc tính boolean này nếu được chỉ định, video sẽ tự động bắt đầu.
<b>autobuffer</b>	Thuộc tính boolean này nếu được chỉ định, video sẽ tự động bắt đầu đưa vào bộ đệm ngay cả khi nó không được thiết lập bật tự động.
<b>controls</b>	Cho phép người sử dụng để kiểm soát phát lại video, bao gồm âm lượng, tìm kiếm, và tạm dừng / tiếp tục phát lại.
<b>height</b>	Thuộc tính này xác định chiều cao của vùng hiển thị của video, theo pixel CSS.

# Multimedia (Tag Video)

```
<!DOCTYPE HTML>

<html>

<body>

    <video width="800" height="500" controls="controls" autoplay="autoplay">
        <source src="E:\Demo.mp4" type="video/mp4">
        <source src="E:\Demo.webm" type="video/webm">
    </video>

</body>

</html>
```

# Multimedia (Tag Audio)

- Nhúng một nội dung âm thanh (audio) vào trang web.

```
<audio controls autoplay>  
  <source src="E:\Demo.mp3" type="audio/mp3">  
  <source src="E:\Demo.ogg" type="audio/ogg">  
</audio>
```

- **"E:\Demo.mp3"** : Đường dẫn đến audio.
- Thẻ `<audio>` có thể có nhiều thẻ `<source>`, `<source>` có thể chứa nhiều audio khác nhau.
- Hỗ trợ 3 định dạng audio: mp3, ogg, wav
- Cần được bổ sung các thuộc tính cho audio, để có thể điều khiển,...được audio

# Multimedia (Tag Audio)

- Thuộc tính kỹ thuật của audio

Thuộc tính	Mô tả
<b>autoplay</b>	Thuộc tính boolean này nếu được chỉ định, âm thanh sẽ tự động bắt đầu chạy
<b>autobuffer</b>	Thuộc tính boolean này nếu được chỉ định, âm thanh sẽ tự động bắt đầu đưa vào bộ đệm ngay cả khi nó không được thiết lập bật tự động.
<b>controls</b>	Cho phép người sử dụng để kiểm soát phát lại âm thanh, bao gồm âm lượng, tìm kiếm, và tạm dừng / tiếp tục phát lại.
<b>loop</b>	Thuộc tính này xác định cho phép âm thanh tự trở lại bắt đầu, sau khi kết thúc.



# Multimedia (Tag Audio)

- Thuộc tính kỹ thuật của audio

Thuộc tính	Mô tả
<b>preload</b>	Thuộc tính này xác chỉ ra rằng âm thanh sẽ được nạp vào tải trang, và sẵn sàng để chạy. Bỏ qua nếu autoplay là hiện tại.
<b>src</b>	Đây là tùy chọn, thay vào đó bạn có thể sử dụng các thẻ <source> trong khối video để xác định video nhúng

# Multimedia (Tag Audio)

- Quản lý sự kiện Media: Thẻ âm thanh và thẻ video HTML5 có thể có một số thuộc tính để điều khiển các chức năng khác nhau của điều khiển bằng cách sử dụng Javascript

Sự kiện	Mô tả
<b>abort</b>	Sự kiện được tạo ra khi phát lại được hủy.
<b>canplay</b>	Sự kiện được tạo khi đủ dữ liệu có sẵn các đa phương tiện có thể được phát.
<b>error</b>	Sự kiện được tạo ra khi một lỗi xảy ra.
<b>ended</b>	Sự kiện được tạo ra khi phát lại hoàn tất.
<b>loadeddata</b>	Sự kiện này được tạo ra khi frame đầu tiên của đa phương tiện đã tải xong.

# Multimedia (Tag Audio)

Sự kiện	Mô tả
<b>loadstart</b>	Sự kiện được tạo ra khi việc tải đa phương tiện bắt đầu.
<b>pause</b>	Sự kiện được tạo ra khi phát lại bị tạm dừng.
<b>play</b>	Sự kiện được tạo ra khi phát lại bắt đầu hoặc phục hồi lại..
<b>progress</b>	Sự kiện được tạo ra theo định kỳ để thông báo sự tiến trình của việc tải về đa phương tiện.
<b>ratechange</b>	Sự kiện này được tạo ra khi thay đổi tốc độ phát lại.
<b>seeked</b>	Sự kiện được tạo ra khi một thao tác tìm kiếm hoàn tất.

# Multimedia (Tag Audio)

Sự kiện	Mô tả
<b>seeking</b>	Được tạo ra khi một hoạt động tìm kiếm bắt đầu.
<b>suspend</b>	Sự kiện này được tạo ra khi tải của đa phương tiện bị đình chỉ.
<b>volumechange</b>	Được tạo ra khi thay đổi âm lượng.
<b>waiting</b>	Sự kiện này được tạo ra khi hoạt động yêu cầu (như phát lại) bị trì hoãn trong khi chờ hoàn thành các hoạt động khác (chẳng hạn như "tìm kiếm").

# Multimedia (Tag Audio)

```
<!DOCTYPE HTML>
<html>
<body>
  <audio controls autoplay loop="loop">
    <source src="E:\Keyshia Cole - Falling Out.mp3" type="audio/mp3" />
    <source src="E:\Keyshia Cole - Falling Out.ogg" type="audio/ogg" />
  </audio>
</body>
</html>
```

# Web Storage

# Web Storage

- Web storage : Cho phép lưu trữ thông tin, dữ liệu bằng trình duyệt web của người dùng.
- Đối tượng của Web storage:
  - localStorage: store data không thời hạn
  - sessionStorage: store data cho một session

**Kiểm tra trình duyệt có hỗ trợ ?**

```
if (typeof (Storage) !== "undefined")  
{  
    // Yes! localStorage and sessionStorage support!  
    // Some code.....  
}  
else {  
    // Sorry! No web storage support..  
}
```

# Web Storage

- Dữ liệu lưu trữ trong Storage chỉ là kiểu chuỗi. Mỗi đối tượng Storage là một danh sách các cặp **key/value**, đối tượng này bao gồm các thuộc tính và phương thức:

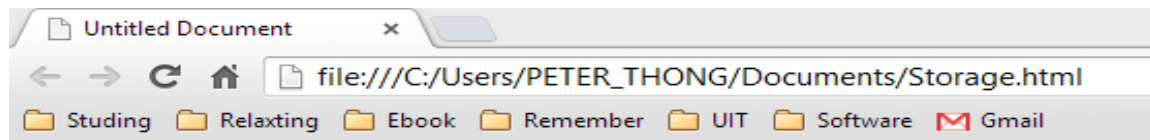
Tên	Mô tả
<b>length</b>	Số lượng cặp key/value có trong đối tượng.
<b>key(n)</b>	Trả về tên của key thứ n trong danh sách.
<b>getItem(key)</b>	Trả về value được gắn với key.
<b>setItem(key,value)</b>	Thêm hoặc gán một cặp key/value vào danh sách.
<b>removeItem(key)</b>	Xóa cặp key/value khỏi danh sách.
<b>clear</b>	Xóa toàn bộ dữ liệu trong danh sách.



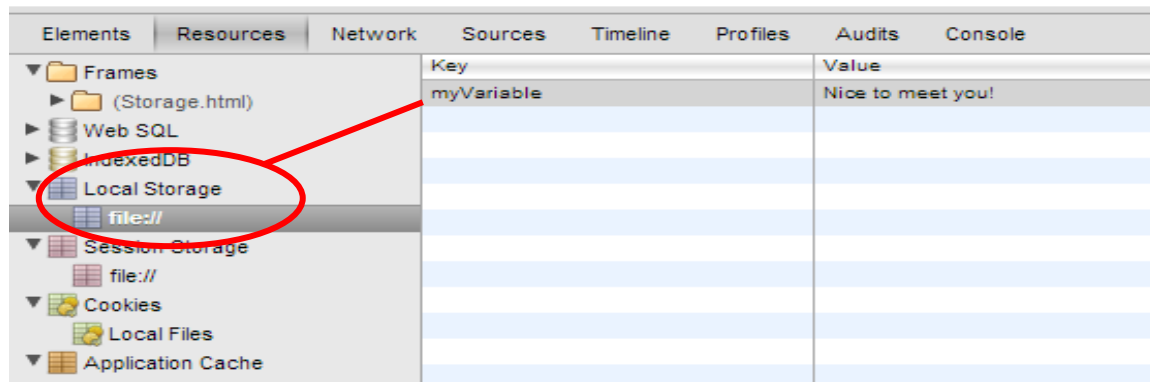
# Web Storage

```
<!DOCTYPE HTML>
<html>
<body>
<script type="text/javascript">
    sessionStorage.myVariable="Hello. ";
    localStorage.myVariable="Nice to meet you!";
    document.write(sessionStorage.myVariable);
    document.write(localStorage.myVariable);
</script>
<body>
</html>
```

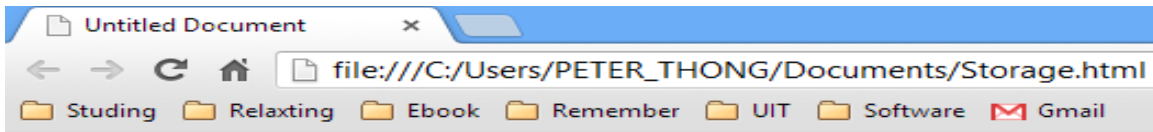
# Web Storage



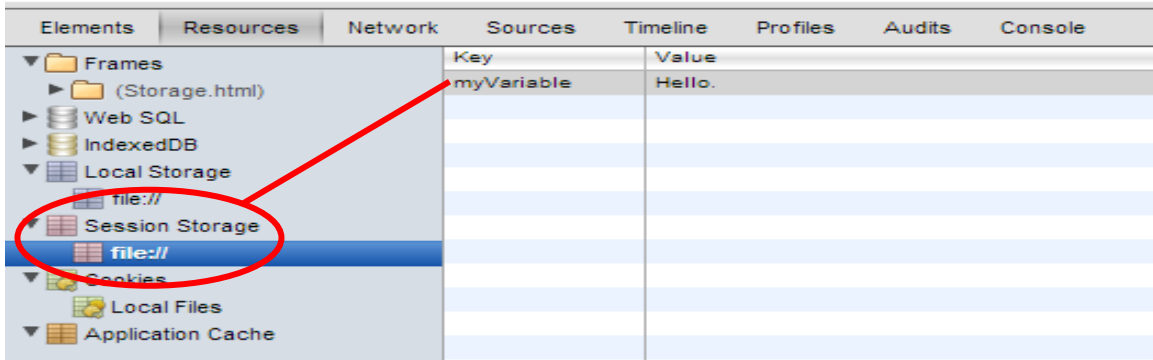
Hello. Nice to meet you!



# Web Storage



Hello. Nice to meet you!





## Cảm ơn đã theo dõi

Hy vọng cùng nhau đi đến thành công.